DOI:
10.1016/j.cogsys.2003.11.002

Document Version
Peer reviewed version

Cognitive Syatems Research, 5(2):93–117, 2004.

Neural Coding Strategies and Mechanisms of Competition

M. W. Spratling and M. H. Johnson
Centre for Brain and Cognitive Development, Birkbeck College, London. UK.

Abstract

A long running debate has concerned the question of whether neural representations are encoded using a distributed or a local coding scheme. In both schemes individual neurons respond to certain specific patterns of pre-synaptic activity. Hence, rather than being dichotomous, both coding schemes are based on the same representational mechanism. We argue that a population of neurons needs to be capable of learning both local and distributed representations, as appropriate to the task, and should be capable of generating both local and distributed codes in response to different stimuli. Many neural network algorithms, which are often employed as models of cognitive processes, fail to meet all these requirements. In contrast, we present a neural network architecture which enables a single algorithm to efficiently learn, and respond using, both types of coding scheme.

Keywords: Representation, Neural Networks, Cerebral Cortex, Competition, Lateral Inhibition

1 Introduction

Understanding how information is encoded in the brain is a fundamental issue for the cognitive sciences. It is generally acknowledged that the activities of neurons carry information (where 'activity' might refer to the firing rate, spike timing, or some other property of the neural response (deCharms and Zador, 2000)). However, controversy persists as to how information is encoded across patterns of activity within populations of neural units. The principle debate on this issue concerns whether information is represented using a distributed or local code (van Gelder, 1999; Thorpe, 1995; Page, 2000; Eggermont, 1998; deCharms and Zador, 2000). Figure 1 illustrates the distinction that is usually made between these coding schemes. For a local code, a single active node represents the current input stimulus and each stimulus is represented by the activation of a distinct node. For a distributed code, stimuli are represented as a pattern of activity across the nodes of the network and unique combinations of active nodes represent each stimulus.

A great deal of discussion has focused on which of these coding schemes is the most likely to be used by the cerebral cortex and hence which is the most appropriate for connectionist models of cognitive processes. Such arguments have focused both on which scheme has the superior computational properties as well as which scheme is most strongly supported by the physiological evidence (Földiák and Young, 1995). In arguing for one coding scheme, to exclusion of the other, both sets of proponents are tacitly assuming that these coding schemes are mutually exclusive. Hence, the assumption is that if one coding scheme can be shown to have computational properties essential to a particular task, or if one cortical region can be shown to generate representations encoded using a certain scheme, then all representations must be of this type. There seems to be little evidence to support such generalization; on the contrary, it seems likely that different representational properties will be required for different cognitive tasks (Markman and Dietrich, 2000). This paper aims to challenge the assumption that local

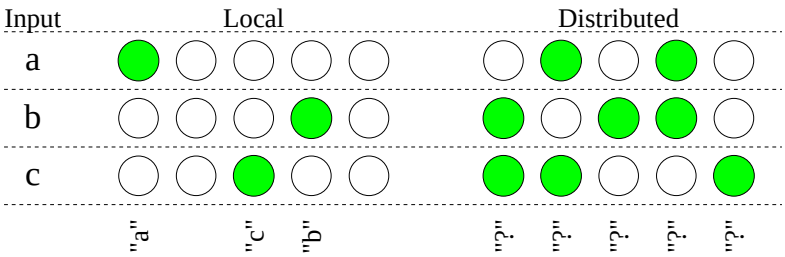

Figure 1: Illustrations of local and distributed coding schemes. Two simple neural networks, each containing five nodes, are shown (nodes are depicted as circles and active nodes are shaded). Each row illustrates the response of the two networks to a different stimulus. The network on the left responds using a local code while the network on the right produces a distributed representation.

and distributed coding schemes are mutually exclusive. There are two ways in which these coding schemes could coexist:

1. Different cortical regions might employ different coding strategies.

2. Each cortical region might employ both types of coding scheme.

The first possibility has been explored in several previous models (Wermter and Sun, 2000; McGarry et al., 1999). Often these models consist of two distinct subsystems, each employing different neural architectures and learning algorithms, to support different types of representation (Sun, 1995, 2001). In this article, we also argue that the same information may be represented using a distributed code in one cortical region and using a local code in another cortical region (see section 4). However, we also make the stronger claim that each cortical region employs both types of coding scheme, and hence, that the same populations of neurons are used to generate local and distributed codes of different stimuli. Furthermore, we argue (in section 2) that neural representations with the properties of both local and distributed codes will be necessary for any population of neurons to accurately and completely represent different stimuli. Hence, any neural network ought to be able to generate both local and distributed representations in response to different stimuli. While this is true for many algorithms, a certain, highly influential, class of competitive (self-organising) neural network algorithms require that all stimuli be represented using a single coding scheme. We argue (in section 3) that such algorithms provide inadequate models of neural representations in the cortex and of cognitive phenomena in general. These deficiencies are overcome with an alternative neural network architecture, that can generate both local and distributed representations as appropriate to the task it was trained on and the current stimulus it is exposed to. Our stance thus contrasts with those that argue that local and distributed coding schemes are distinct (*e.g.*, van Gelder, 1999; Thorpe, 1995), those that propose that these schemes are mutually exclusive (*e.g.*, Földiák and Young, 1995; O'Reilly, 1998), and those that propose that distinct neural architectures are necessary for each type of code (*e.g.*, Sun, 1995, 2001).

2 Neural Coding Strategies

There are two dimensions along which neural coding strategies are commonly distinguished: the tuning properties (or stimulus selectivities) of the individual neurons and the number of nodes that are active in representing a single stimulus. In this section, we argue that these properties fail to provide a clear distinction between local and distributed codes. Rather than identifying a qualitative difference between distinct coding schemes, these properties describe two types of code that may be generated by a single underlying representational mechanism. Hence, the terms "local" and "distributed" describe properties of a single coding mechanism, rather than identifying two distinct, mutually exclusive, coding schemes.

2.1 Tuning Properties

A standard example of a distributed representation is the binary ASCII code. In this code, any individual bit (or node) will be ON as part of the representation of a number of different alphanumeric characters. In contrast, using a localist representation, each character would commonly be represented by a distinct node. If such a localist network was used to process visual images of characters then we might expect that a single node would respond to many different images of the same character presented in different fonts, or in different handwriting, or in different locations within the image. Similarly, we would expect the archetypal 'grandmother cell' to respond to very many different views of grandmother. For both the ASCII code example and for the grandmother cell representation the required tuning properties of individual nodes are identical: in both cases an individual node selectively responds to a number of specific stimuli.

Cells with other response properties are also equally capable of generating either local or distributed codes. For example, neural network models commonly employ nodes which have an activation calculated as the weighted sum of the input activities passed through a nonlinearity (such as the sigmoid function). Such nodes will respond most strongly to a single specific stimulus, although they may be broadly tuned and thus respond with varying strength over a range similar patterns of input activity. A network of nodes using this activation function can generate a local code if individual nodes respond most strongly to entire input patterns. Such a network could alternatively generate a distributed code if nodes respond to sub-features of each stimulus.

It is therefore not the case that nodes with particular tuning properties necessarily generate a local or a distributed code. Hence, such coding schemes cannot be distinguished solely by examining the response properties of neurons. Such a conclusion should come as no surprise, since neural network models, generating each type of code, use identical activation functions. However, tuning properties have been claimed as a distinguishing feature of coding schemes (Thorpe, 1995; Page, 2000; Barlow, 1995). It is claimed that a local code is characterized as

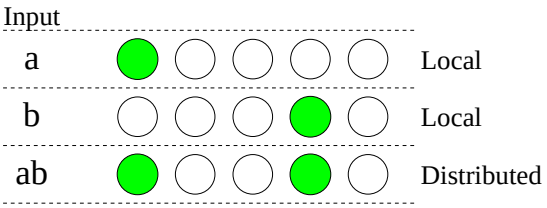

Figure 2: An illustration that the coding strategy employed should depend on the stimulus and not be a predefined property of the neural network. A simple neural network, containing five nodes, is shown (nodes are depicted as circles and active nodes are shaded). Each row illustrates the response of the network to a different stimulus. The network needs to respond using a different number of active nodes depending on the stimulus, and hence the same network needs to generate local and distributed codes in different circumstances.

one in which the preferred stimulus of each individual node has a clear interpretation to which we can apply a meaningful label, while a distributed code is characterized as one in which the activity of an individual node in isolation does not have a clear meaning (Thorpe, 1995; Page, 2000; Barlow, 1995). By this definition, if we invent words to describe the preferred stimuli of nodes participating in generating a distributed code, then the representation becomes a locally encoded. Hence, at best such a definition is arbitrary, depending on what is considered meaningful (Hummel, 2000) or which categories have a readily available linguistic label. At worst this distinction is simply incorrect as it does not hold true in general; for example, distributed representations, in which individual nodes are tuned to meaningful sub-features, are common and have been labelled featural codes (Page, 2000).

Neurons respond to certain, specific patterns of pre-synaptic activity. This is true when neurons are participating in a local code and when neurons are participating in a distributed code. Hence, rather than being dichotomous, both coding schemes are based on the same representational mechanism. In some cases the patterns of pre-synaptic activity to which a neurons respond can be given meaningful labels, however, this is irrelevant to the process by which information is encoded in the neural activity.

2.2 Number of Active Nodes

Another property that is often suggested as a distinguishing feature of coding schemes, is the number of nodes that are active in response to an individual input pattern. A local code is characterized as one in which a single node is active, while a distributed code is characterized as one in which more than one node is simultaneously active (Thorpe, 1995; Page, 2000; van Gelder, 1999). This definition is far from unambiguous (Thorpe, 1995). For example, consider a system containing two distinct neural populations, in both of which a single neuron is active. This could be considered as two local representations or one distributed representation. The same ambiguity arises for a single population of neurons when members of that population have distinct, non-overlapping, receptive fields. Furthermore, if a single network contained two neurons with identical synaptic weights, when these two neurons (and only these two neurons were active) this might be considered as a more robust, redundant, local code (Thorpe, 1995). However, if the number of active neurons distinguishes coding schemes, then this must be classified as a distributed code.

Consider a neural network in which each individual node represents a single letter of the alphabet. When a single letter is presented to this network a single node should respond. When two (or more) letters are simultaneously presented, two (or more) nodes should be active in order to completely represent the input stimulus (see figure 2). Hence, if the number of nodes that are active in response to a stimulus distinguishes a local code from a distributed code, then this network would be described as generating a local code to single-letter stimuli and a distributed code to multiple-letter stimuli. However, nothing about the network or the way in which it responds to stimuli has changed. This distinction thus appears to be fairly arbitrary.

The above example illustrates that for any neural network it is always possible to devise a stimulus that matches the selectivity of an individual node, and hence ought to cause the activation of one node. Conversely, it is also always possible to devise a stimulus that matches the selectivities of several nodes, and hence ought to cause the activation of multiple nodes. The ability to be able to represent all the features of the stimulus, and only those features, regardless of how many active nodes are required to do so, is essential if a neural network is to accurately represent stimuli. Ideally, any neural network should thus be capable of generating both local and distributed codes in response to different stimuli. Furthermore, this example illustrates that it is wrong to assert that all stimuli must be represented using a single coding scheme. However, this is the claim that is often made by both

sets of proponents in the debate about local and distributed coding.

For any neural network the synaptic weights of the nodes together with the current stimulus properties should determine which of these nodes respond and hence how many nodes are active. It is thus essential when describing the coding scheme employed by a neural network to specify the stimuli for which it generates such a code. It is then appropriate to refer to a specific stimulus being represented locally (by the activation of a single node) or being represented in a distributed manner (by the pattern of activity across a number of nodes). We will thus continue to use these terms in this way: as convenient labels for a property of a single coding mechanism, rather than as labels for distinct coding schemes. Using this definition a single neural network may be described as representing some stimuli using a distributed code and other stimuli using a local code. A network that produces a distributed representation of a particular stimulus may generate a local representation of a sub-feature of that same stimulus. We might even consider that the network produces both of these representations simultaneously: providing a distributed representation of the entire stimulus while generating many localist representations of the sub-features. The tuning properties of neurons result from the activation function that is employed and the synaptic weights over which this function operates. Since the strengths of synaptic weights are often learnt, the training data and learning rules are also important factors in determining the encoding used to represent the current stimulus.

3 Mechanisms of Competition

In the previous section we have argued that rather than being mutually exclusive or independent coding schemes, both local and distributed codes are based on the same representational mechanism in which individual neurons encode information based on the selectivity of their synaptic weights. Whatever the selectivities of the nodes, both local and distributed codes will be necessary to accurately and completely represent different stimuli (since for any given network it is possible to devise a stimulus that can only be correctly represented by the activation of a single node, and to devise a stimulus that can only be correctly represented by the activation of multiple nodes). Hence, any neural network ought to be able to generate both local and distributed representations in response to different stimuli. In addition, any neural network should have the ability to learn both distributed and local representations as appropriate to the task. In this section, we argue that a class of conventional neural network models fail to meet these requirements. We also describe an alternative algorithm which does meet these criteria.

3.1 Post-Integration Lateral Inhibition

Numerous neural network algorithms employ a common mechanism for generating competition between nodes. Nodes compete for the right to be active in response to the current stimulus. A node's success in this competition is dependent on the total strength of the stimulation it receives, and nodes which compete unsuccessfully have their output activity suppressed. This form of competition can be implemented either by using a selection process which chooses the 'winning' node(s) (*e.g.*, Rumelhart and Zipser, 1985; Kohonen, 1997; Grossberg, 1987; Ritter et al., 1992; Földiák, 1991; Wallis, 1996; O'Reilly and Munakata, 2000) or, as illustrated in figure 3(a), by using lateral inhibitory connections between nodes (*e.g.*, Földiák, 1989, 1990; Marshall, 1995; Sirosh and Miikkulainen, 1994; von der Malsburg, 1973; Oja, 1989; Sanger, 1989; Murre et al., 1992). We describe this class of neural networks as employing 'post-integration lateral inhibition'.

Post-integration lateral inhibition has been used to generate representations in which single nodes are active at any one time as well as codes in which multiple nodes participate in representing a single stimulus. To produce a local code a selection mechanism can be used which chooses a single winning node to be active in response to each stimulus (*i.e.*, winner-takes-all competition). Alternatively, if explicit inhibitory lateral connections are employed, then these weights simply need to be sufficiently strong to allow the activity of a single node to be dominant at any one time. Both these implementations allow only a single active node, and hence will generate a local code irrespective of the actual input stimulus. To produce a distributed code, a selection mechanism can be used which chooses multiple winning nodes (*i.e.*, k-winners-take-all competition). Such a mechanism will ensure that k nodes are active in response to every stimulus, and hence will force a distributed representation to be generated irrespective of the actual input stimulus. Such strong constraints on the number of active nodes will prevent proper expression of the knowledge stored in the synaptic weights and necessarily means that these networks will be incapable of accurately representing certain stimuli. Furthermore, such algorithms are unable to cope with tasks that require an arbitrary number of active nodes: even tasks as simple as responding to single or multiple letters (as illustrated in figure 2).

Networks which employ explicit lateral connections, offer greater representational flexibility, since the strength of competition between every pair of nodes can be different. However, exploitation of these extra degrees of freedom requires that weights can be specified which are appropriate to the current task and desired encoding.

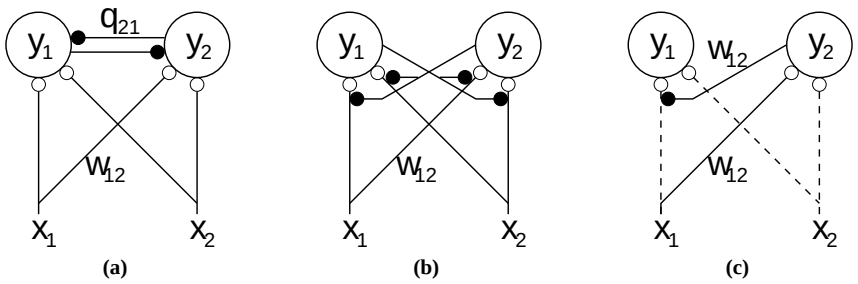

Figure 3: Mechanisms of lateral inhibition. Nodes are shown as large circles, excitatory synapses as small open circles and inhibitory synapses as small filled circles. (a) The standard model of lateral inhibition provides competition between outputs. (b) The pre-integration lateral inhibition architecture provides competition for inputs. (c) The pre-integration lateral inhibition model with only one lateral weight shown. This lateral weight has an identical value (w_{12}) to the afferent weight shown as a solid line.

Independently tuned lateral weights could be obtained by specifying separate, task specific, learning rules for every connection. However, in practice, a single learning rule is used to adjust the weight of every lateral connection. For example, Földiák (1990) used the following equation: $\Delta q_{kj} = \beta \left(y_j y_k - p^2 \right)$, where β is a parameter controlling the rate of learning, q_{kj} is the lateral weight from node k to node j, y_j and y_k are the activities of these two nodes, and p^2 is a parameter which controls how often these two nodes will be coactive. Irrespective of the task, this learning rule will always force the network to generate a distributed encoding of a proportion of the training data. The choice of the value of parameter p^2, or the choice a different learning rule, constitutes an *a priori* choice of how the training data will be encoded. Hence, although these models can generate both local and distributed representation in response to different stimuli, they are incapable of flexibly learning different encodings for different tasks.

From the above discussion it can be seen that neural network algorithms that use post-integration lateral inhibition as the mechanism for competition require an *a priori* choice to be made as to whether the training data will be represented using a local or distributed code. Any one algorithm will produce the same coding scheme for every task it is applied to, irrespective of the properties of the particular training data. For this reason it has become common practice to describe a neural network as generating a local or a distributed code. Furthermore, the requirement that different algorithms be used to produce different coding schemes may have contributed to the idea that a neural network, and hence the cerebral cortex, is committed to using either a local or a distributed code. However, rather than identifying a true dichotomy between coding schemes this requirement actually results from a weakness of these post-integration lateral inhibition algorithms.

3.2 Pre-Integration Lateral Inhibition

The tuning properties of the nodes should determine which nodes, and hence how many nodes, will be active in response to a particular stimulus. To achieve this requires a mechanism of competition which does not impose constraints on the number of nodes that should be simultaneously active. Furthermore, the mechanism of competition should allow different coding schemes to be learnt for different tasks. We present a simple, biologically plausible, neural network architecture which achieves these aims. With this algorithm, competition is provided by each node 'blocking' its preferred inputs from activating other nodes (see figure 3(b)). Nodes thus compete for the right to receive inputs rather than for the right to generate outputs. We describe this form of competition as 'pre-integration lateral inhibition'. The algorithm is described fully in the Appendix, and is justified on both biological and computational grounds in Spratling and Johnson (2002). In that paper we demonstrated the efficiency (in terms of speed and reliability) with which a neural network employing pre-integration lateral inhibition can learn. In the present paper we explore the types of neural code that can be generated by such a network and demonstrate the improved representational properties of this architecture in comparison to post-integration lateral inhibition models.

For pre-integration lateral inhibition the magnitudes of the lateral weights are identical to the corresponding afferent weights. For example, as illustrated in figure 3(c), the activation of node j inhibits input i of node 1 with a strength of w_{ij}. This lateral weight is identical to the afferent weight node j receives for input i. With this arrangement of weights, if a node is strongly activated by the overall stimulus and it has a strong synaptic weight to a certain feature of that stimulus, then it will inhibit other nodes from responding to that feature. On

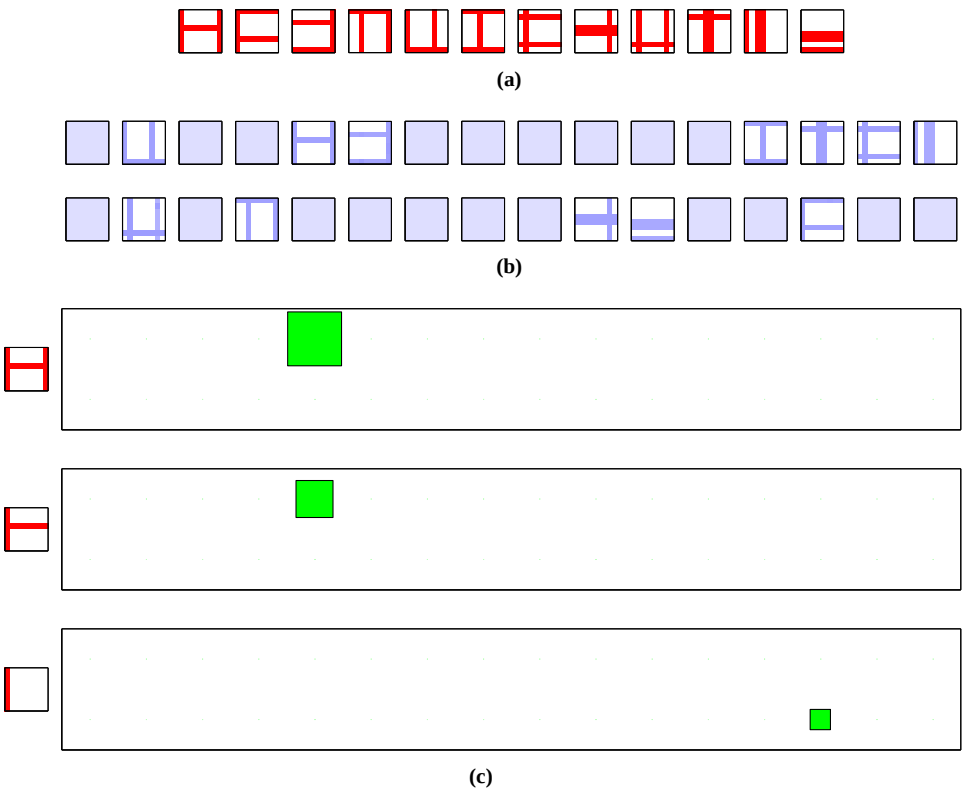

Figure 4: A network generating a local code for three bar patterns. (a) Training data consisting of 12 images each on an 8 by 8 grid of pixels. Dark pixels indicate active inputs. (b) The synaptic weights for the 32 nodes in a network after training for 384 iterations. Each box represents the weights learnt by an individual node. The darkness of the pixels within each box corresponds to the strength of that weight. (c) The response of the network, after training, to three test patterns. The left-most column shows the input pattern, with the remainder of each row showing the response of the network to this pattern. The size of each square corresponds to the strength of the response of each node. Responses are shown in the same order as the weights are shown in (b). This network generates a local code in response to one of the training patterns.

the other hand, if an active node receives only weak weights from a feature then it will only weakly inhibit other nodes from responding to that feature. Hence, if individual nodes have learnt weights that are selective to certain stimuli then when multiple stimuli are simultaneously presented to the network each of the nodes representing one of these stimuli can be simultaneously active. However, there remains strong competition between nodes which are selective for overlapping input patterns and hence only those nodes which have receptive fields that most closely match the stimulus will win the competition and remain active. Since the lateral weights have identical values to afferent weights it is not necessary to determine their strengths separately. It is simply necessary to learn afferent weights which are appropriate to the given task and these weights automatically generate suitable competition between the nodes. This results in efficient learning (Spratling and Johnson, 2002). Furthermore, this algorithm has no more degrees of freedom than one employing post-integration lateral inhibition, and hence can be legitimately compared to such models.

In contrast to post-integration lateral inhibition models, in which the mechanism of competition strongly influences the tuning properties that are learnt by the nodes, in this model learning is purely dependent on the learning rule for the afferent weights. For a given learning rule the features that are learnt by individual nodes in a network will depend on the data that is presented during training. We illustrate this by showing the results of using different sets of training data with such a network using the learning rule described in the Appendix. Stimuli consisted of eight by eight pixel images. Each input pattern was generated by activating a fixed number of horizontal or vertical bars within the image. Using patterns consisting of three bars it was found that a training set of 12 or fewer images resulted in individual nodes becoming selective to the training patterns themselves (an example is shown in figure 4). The network thus generated a local code in response to each image in the training set. It can be seen in figure 4(b) that each node learns weights that correspond to one of the training patterns and when

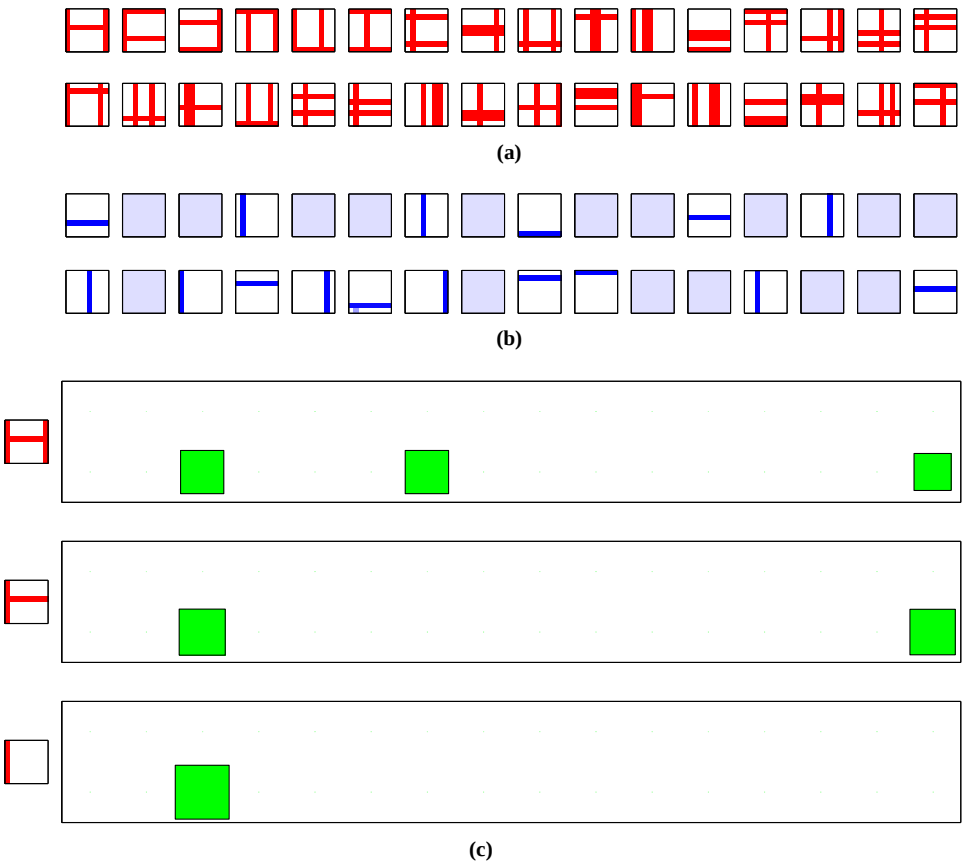

Figure 5: A network generating a distributed code for three bar patterns. (a) Training data consisting of 32 images each on an 8 by 8 grid of pixels. Dark pixels indicate active inputs. (b) The synaptic weights for the 32 nodes in a network after training for 1024 cycles. Each box represents the weights learnt by an individual node. The darkness of the pixels within each box corresponds to the strength of that weight. (c) The response of the network, after training, to three test patterns. The left-most column shows the input pattern, with the remainder of each row showing the response of the network to this pattern. The size of each square corresponds to the strength of the response of each node. Responses are shown in the same order as the weights are shown in (b). This network generates a distributed code in response to one of the training patterns and a local code in response to an isolated bar feature.

presented with one of these patterns a single node becomes fully active (figure 4(c)). When tested with partial patterns single nodes respond with reduced strength. These results were consistent across trials. The network was trained 25 times using different randomly generated sequences of the twelve input patterns. For each of these trials the network learnt a local encoding of the training data, with the majority of networks doing so within 240 training cycles. Weights were generally learnt after one presentation of each training pattern and remained stable with further training.

For larger training sets individual nodes became selective to sub-features that were common to multiple stimuli. Using a training set consisting of 32 or more images individual nodes learnt weights selective for individual bars (an example is shown in figure 5). It can be seen in figure 5(b) that each node learns weights that correspond to one of the 16 possible horizontal and vertical bars which make up all the training patterns. Hence, the network generates a distributed representation, with three active nodes, when presented with one of the training patterns (figure 5(c)). These results were consistent across trials. The network was trained 25 times using different randomly generated sequences of the 32 input patterns. For each of these trials the nodes became selective to individual bars, with the majority of networks doing so within 430 training cycles. The trained network can also successfully represent other patterns consisting of arbitrary numbers of bar features. Hence, when tested with partial patterns each node, that is selective to a bar in the test pattern, became active (figure 5(c)). The number of active nodes is thus dependent on the current stimulus, and different stimuli are represented by either a local or a distributed code. Note that the network contains sufficient nodes to be able to represent each of the training patterns using an individual node. However, after training the network contains many uncommitted nodes, and the

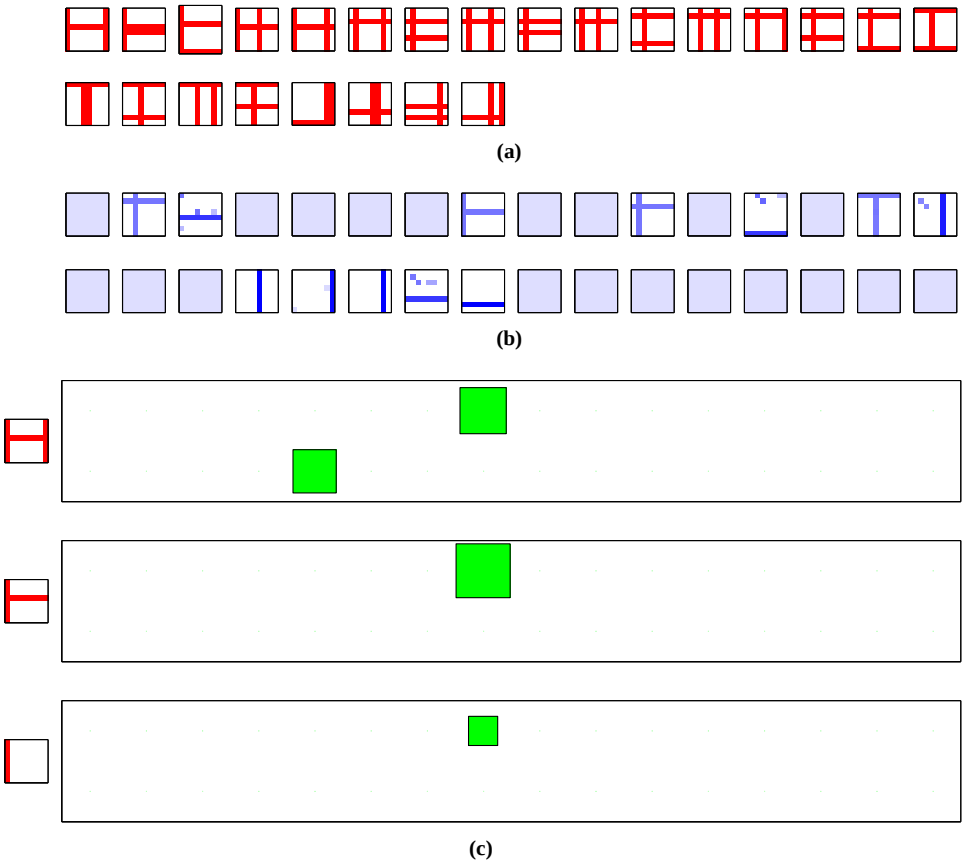

Figure 6: A network generating a distributed code for three bar patterns based on nodes selective for different sub-features to figure 5. (a) Training data consisting of 24 images each on an 8 by 8 grid of pixels. Dark pixels indicate active inputs. (b) The synaptic weights for the 32 nodes in a network after training for 768 cycles. Each box represents the weights learnt by an individual node. The darkness of the pixels within each box corresponds to the strength of that weight. (c) The response of the network, after training, to three test patterns. The left-most column shows the input pattern, with the remainder of each row showing the response of the network to this pattern. The size of each square corresponds to the strength of the response of each node. Responses are shown in the same order as the weights are shown in (b). This network generates a distributed code in response to one of the training patterns and a local code in response to certain pairs of bars and also to certain isolated bar features.

network has learnt to represent all the patterns in the training set using a distributed code. Nodes learn to represent sub-features of the training patterns not because of limitations in resources but because each of these sub-features recurs as part of a number of different patterns. The sub-features are therefore perceived by the learning algorithm as being independent features of the training data.

The features that are learnt by individual nodes depend on the training data and it is not necessary that each node learns to detect a similar feature. Hence, when trained on a suitable set of images, different nodes can learn to become selective to different types of feature. For example, when trained on the images shown in figure 6(a) some nodes in the network learnt to detect individual bars while others became selective to bar pairs. The individual bars are the four horizontal bars from the bottom-half of the image and the four vertical bars from the right-half of the image. The four bar pairs consist of one horizontal and one vertical bar from the top- and left-halves of the image. It can be seen that all the patterns in the training set are made up of either one of these bar pairs together with one of the individual bars, or three of the individual bars.

Similar results are obtained when the training data contains patterns consisting of four bars. With a small training set individual nodes learn to become selective to the training images themselves (*e.g.*, figure 7), with larger training sets nodes learn to be selective to the independent features which make up all the training images, be they individual bars (*e.g.*, figure 8), or whatever other sub-features are appropriate to represent all the images upon which they have been trained (*e.g.*, figure 9).

Algorithms that use post-integration lateral inhibition could be used to learn similar synaptic weights to those

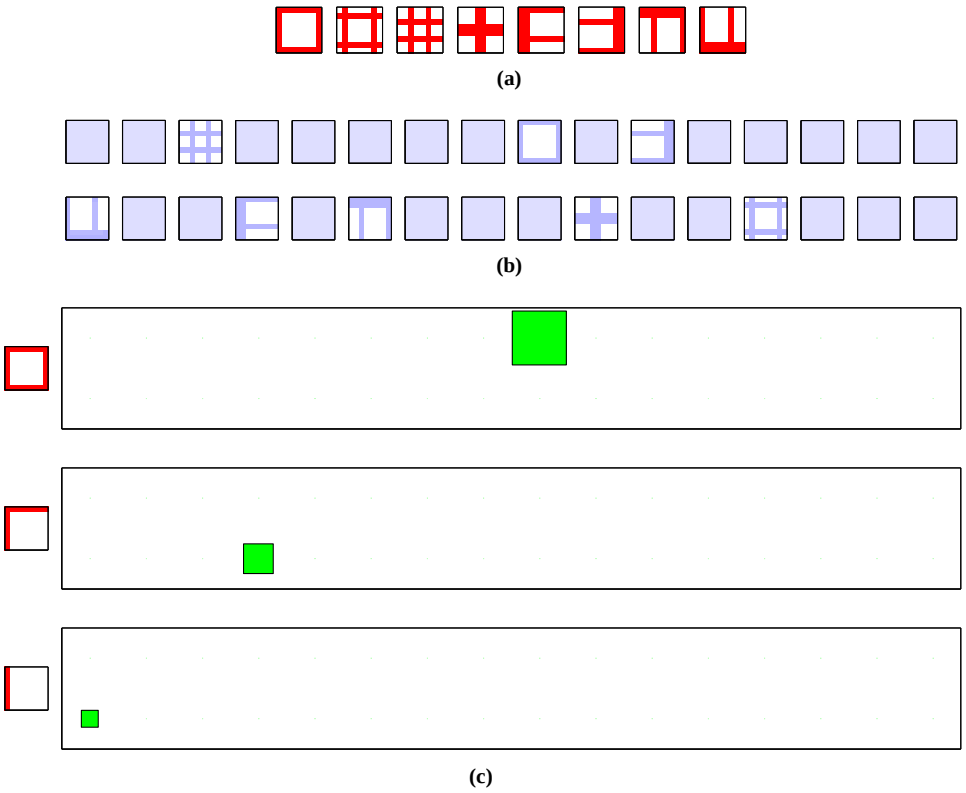

Figure 7: A network generating a local code for four bar patterns. (a) Training data consisting of eight images each on an 8 by 8 grid of pixels. Dark pixels indicate active inputs. (b) The synaptic weights for the 32 nodes in a network after training for 192 iterations. Each box represents the weights learnt by an individual node. The darkness of the pixels within each box corresponds to the strength of that weight. (c) The response of the network, after training, to three test patterns. The left-most column shows the input pattern, with the remainder of each row showing the response of the network to this pattern. The size of each square corresponds to the strength of the response of each node. Responses are shown in the same order as the weights are shown in (b). This network generates a local code in response to one of the training patterns.

presented above, however, a different algorithm would be required for each task. For example, a winner-takes-all algorithm could generate nodes with similar synaptic weights to those shown in figures 4 and 7, and a k-winners-take-all algorithm could learn the synaptic weights shown in figures 5 and 8 with k equal to three in the first case and four in the latter. Similarly, with different values for the parameter p^2, appropriate to each task, the algorithm presented in Földiák (1990) could also produce similar results. However, no single post-integration lateral inhibition algorithm can learn all of these representations.

Many other neural networks algorithms can also learn both distributed and local representations of the training data in different circumstances. Commonly, the capacity of the network (*i.e.*, the number of nodes) in relation to the complexity of the task will determine whether the algorithm learns a local or a distributed encoding of the training data. For example, a multilayer perceptron network trained using back-propagation of error, will generally learn a distributed code of each training pattern in its hidden layer. However, with sufficient nodes in the hidden layer it is also possible for it to learn local representations of the training patterns. For such an algorithm, the encoding of the training data is thus a consequence of an *a priori* choice made by the modeller. In contrast, for the pre-integration lateral inhibition algorithm, described here, how the training data is encoded is dependent on the statistics of the training data. Individual nodes learn to become selective to stimuli that recur within the training data. When such recurring stimuli correspond to entire training patterns, the network learns a code which represents the training patterns using a local code (as shown in figure 4). When the recurring stimuli correspond to sub-features of the training patterns, the network learns a code which represents the training patterns using a distributed code (as shown in figure 5). In the latter example, there are a sufficient number of nodes to learn a local code of the training data. The network learns a distributed code, not due to lack of capacity, but due to the repeated occurrence of the same sub-features across many training patterns.

Figure 8: A network generating a distributed code for four bar patterns. (a) Training data consisting of 64 images each on an 8 by 8 grid of pixels. Dark pixels indicate active inputs. (b) The synaptic weights for the 32 nodes in a network after training for 1536 cycles. Each box represents the weights learnt by an individual node. The darkness of the pixels within each box corresponds to the strength of that weight. (c) The response of the network, after training, to three test patterns. The left-most column shows the input pattern, with the remainder of each row showing the response of the network to this pattern. The size of each square corresponds to the strength of the response of each node. Responses are shown in the same order as the weights are shown in (b). This network generates a distributed code in response to one of the training patterns and a local code in response to an isolated bar feature.

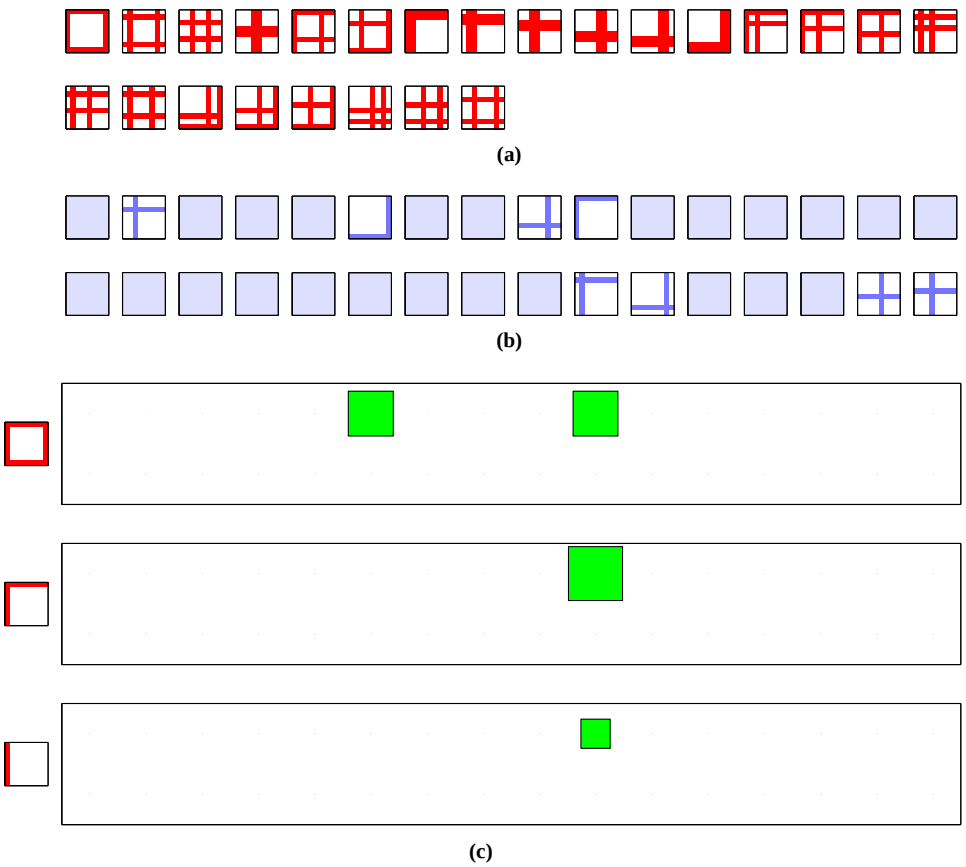

Figure 9: A network generating a distributed code for four bar patterns based on nodes selective for different sub-features to figure 8. (a) Training data consisting of 24 images each on an 8 by 8 grid of pixels. Dark pixels indicate active inputs. (b) The synaptic weights for the 32 nodes in a network after training for 576 cycles. Each box represents the weights learnt by an individual node. The darkness of the pixels within each box corresponds to the strength of that weight. (c) The response of the network, after training, to three test patterns. The left-most column shows the input pattern, with the remainder of each row showing the response of the network to this pattern. The size of each square corresponds to the strength of the response of each node. Responses are shown in the same order as the weights are shown in (b). This network generates a distributed code in response to one of the training patterns and a local code in response to certain pairs of bars and also to certain isolated bar features.

4 Hierarchical Representation

In the simulations presented above, we have demonstrated that a single neural network, employing an appropriate competitive mechanism, can learn and generate both local and distributed codes. In this model, individual neurons become active in response to particular patterns of pre-synaptic activity. Such patterns of activity can not only be provided by input data, but also by the activities of other neural populations. The output of one such neural network could thus provide (at least part of) the input to another network, to form a hierarchical arrangement of neural populations. To illustrate this, we experimented with a hierarchy of two neural networks. One network, consisting of 32 nodes, received sensory input from an eight by eight pixel image. The nodes in this lower-level network had small receptive fields which received afferent connections from only one quadrant of the input space (eight nodes received input from each quadrant). Nodes in the higher-level network received connections from all the nodes in the lower-level network. These nodes thus had larger receptive fields and could learn to become selective to larger, more complex, patterns across the whole of the input space. The selectivities of the nodes in both networks were learnt using the same algorithm as used above.

Figure 10(a) shows typical examples of the training data used in this experiment. Each input pattern consists of one of the images shown in figure 10(b) together with additional bar segments which are selected to be active at random. When trained with such data, nodes in the lower-level network, become tuned to short horizontal and vertical bars, within each quadrant of the image (figure 10(c)). Nodes in the higher-level network become tuned to recurring patterns of activity across the outputs of the lower-level network (figure 10(d)). Hence, nodes in the higher-level network become selective to the patterns shown in figure 10(b). These results were consistent across trials. The hierarchy was trained 10 times using different randomly generated sequences of input patterns. For 70% of these trials the nodes in the higher-level network became selective to each of the ten patterns, with the majority of networks doing so within 90 training cycles.

Figure 11 shows the responses of both networks to a number of input patterns. Both networks within the trained hierarchy represent the same information, but may do so using different numbers of active nodes. Hence, while in previous sections we have argued that a single population of neurons should generate both local and distributed codes in response to different stimuli, here we have demonstrated that multiple populations of neurons can generate both types of code in response to a single stimulus. In the following sections, we argue that the cerebral cortex also employs both local and distributed codes.

4.1 Cortical Coding

The type of stimulus to which a cortical neuron is responsive varies between cortical regions. Cells in primary sensory areas represent low-level features of the sensory input, while cells in higher cortical regions integrate information and respond to more elaborate or abstract features of the environment (Crick and Asanuma, 1986). For example, along the ventral visual pathway (Ungerleider, 1996; Ungerleider and Mishkin, 1982; Goodale and Milner, 1992) cells in the primary visual cortex (V1) are responsive to simple stimuli such as oriented edges at specific locations (Hubel, 1988), while cells in inferotemporal cortex (IT) are selective to complex objects, such as faces, anywhere in the visual field (Perrett et al., 1992; Tanaka, 1996; Logothetis and Sheinberg, 1996; Perrett, 1996). Due to these increasingly complex tuning properties, information represented by the response of many neurons at one level may be represented by the response of an individual neuron at a higher level. Furthermore, neurons which are selective to more abstract stimuli are tuned to patterns of activity across the populations of neurons in more peripheral cortical areas from which they receive their inputs. Such observations have formed the basis for many hierarchical neural network models of perceptual skill learning (*e.g.*, Fukushima, 1988; Riesenhuber and Poggio, 1999). Cells selective to abstract stimuli in one cortical region are built upon distributed representations in lower-level cortical regions. Hence, a local representation, of an abstract feature, does not occur without a distributed representation of the same information. Within a single cortical region a cell which provides a localist representation of a certain stimulus feature contributes to a distributed representations of the stimulus as a whole. Hence, a distributed representation does not occur without localist representations. Far from being mutually exclusive both local and distributed representations coexist in the cortical hierarchy (see figure 12).

Individual cells in V1 signal the presence of simple image features. However, the population of V1 cells must also provide a representation of the visual scene as a whole, and all stimuli that can be distinguished by the visual system must be represented within V1. Since all visual information is available in V1 (and for that matter in the LGN and at the retina) it must be advantageous to recode information so that certain stimuli, represented by the activity of populations of cells in one area, are represented by individual cells in a higher cortical area (*cf.*, Karmiloff-Smith, 1994). Distributed and local codes have complementary computational advantages and disadvantages (Földiák and Young, 1995). For this reason it has been suggested that the cortex uses a sparse

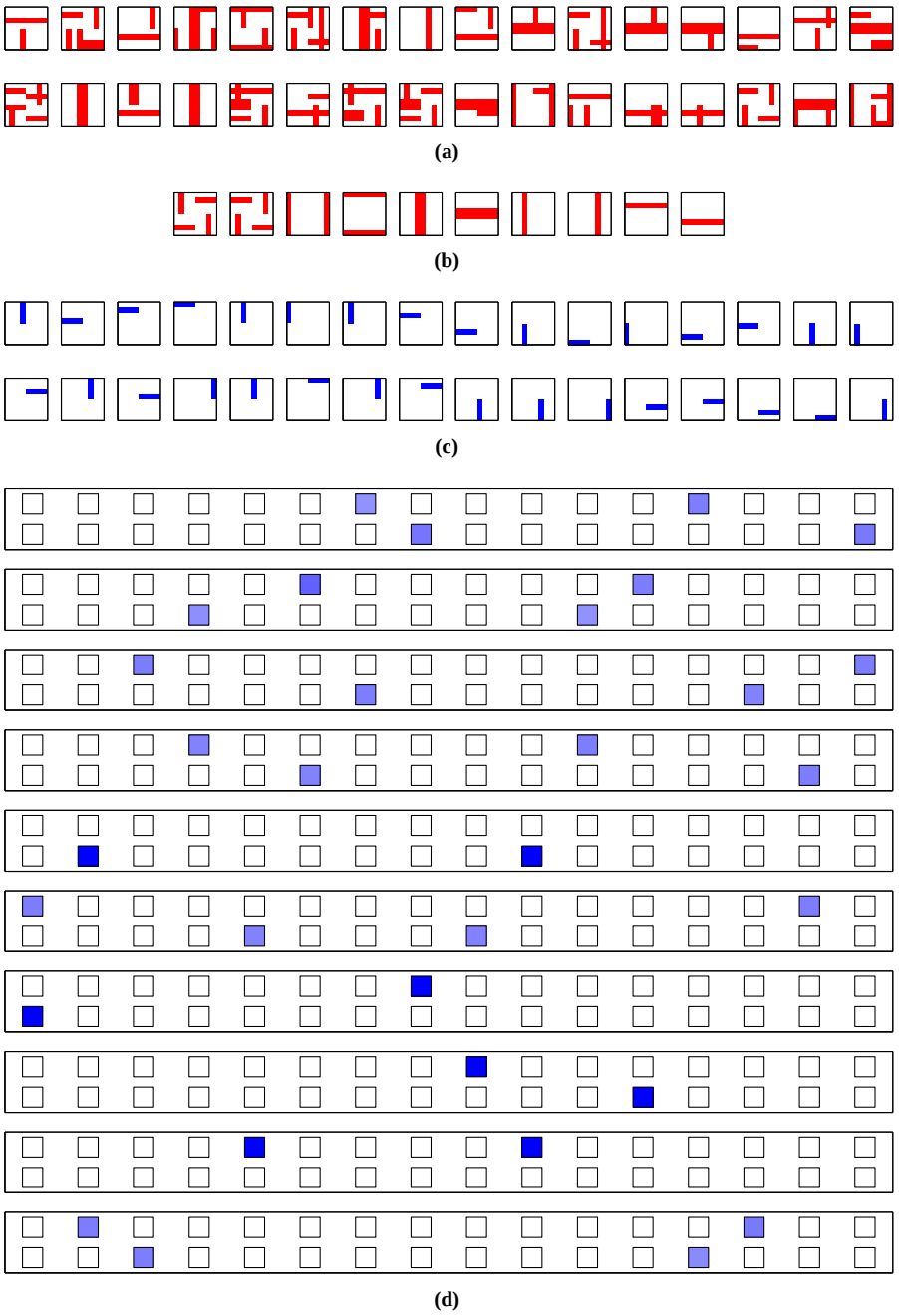

Figure 10: A hierarchy of two networks which learn information at different levels of abstraction. (a) Examples of typical input images used as training data. Each image is an 8 by 8 grid of pixels with dark pixels indicating active inputs. Images are composed of one of the ten patterns shown in (b) together with other bar segments selected which are active with probability 0.04. The synaptic weights, learnt after 500 cycles, for (c) the 32 nodes in the lower-level network, and (d) the 10 nodes in the higher-level network. For each node in the higher-level network, the weights received from each of the 32 nodes in the lower-level network are shown in the same order as these nodes are presented in (c). The darkness of each pixel corresponds to the strength of that weight. Nodes in the lower-level network have become tuned to sets of pixels which correspond to bar segments within the training images, while nodes in the higher-level network have become tuned to sets of nodes in the lower-level network, which are in turn tuned to bar segments that make up the patterns shown in (b).

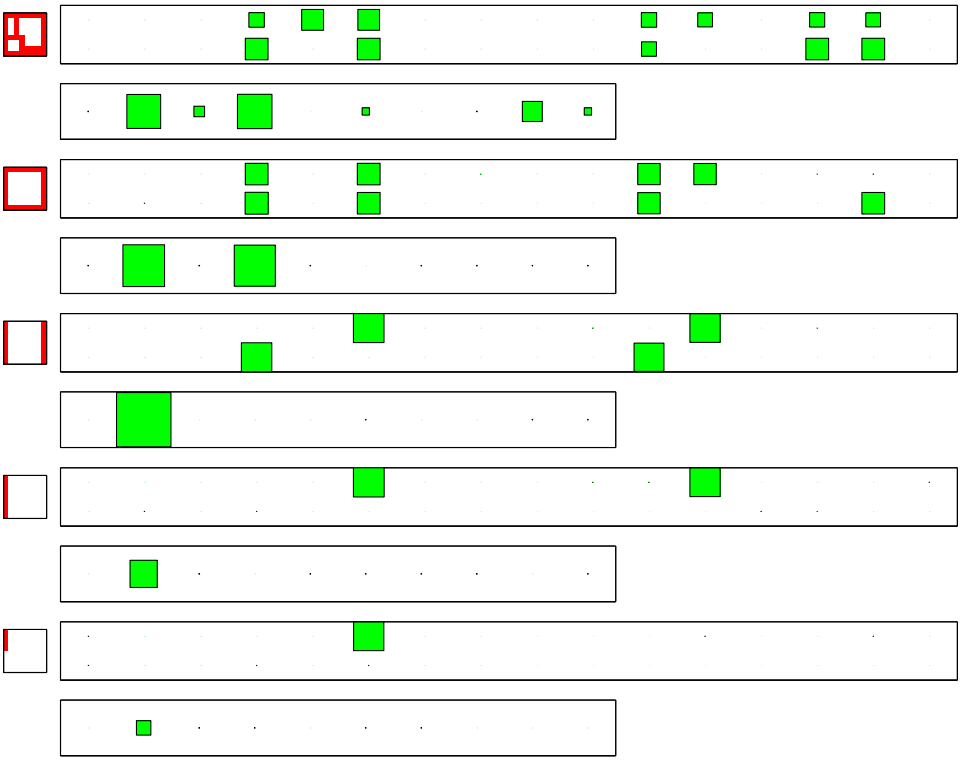

Figure 11: The response of the network shown in figure 10, to five test patterns. The left-most column shows the input pattern, with the remainder of each row showing the response of the network to this pattern. The size of each square corresponds to the strength of the response of each node. Responses for the 32 nodes in the lower-level network are shown in the same order as the weights for these nodes are shown in figure 10(c). Responses for the 10 nodes in the higher-level network are shown (below those for the lower-level network) in the same order as the weights for these nodes are shown in figure 10(d) with the left most response corresponding to the node shown at the top of figure 10(d). For the first two test patterns, both the higher and lower-level networks generate distributed codes. For the third to fifth, test pattern the higher-level network generates a local code.

coding scheme (in which the number of active neurons is intermediate between a local and a fully distributed code) since this coding density offers the best compromise between the computational advantages and disadvantages of local and densely distributed coding (Földiák and Young, 1995; Field, 1994; Feldman, 1990; Olshausen and Field, 1996b,a; Barlow, 1995). However, the cortex need not make such a compromise. A hierarchy of neural networks that represent the same information across a range of levels of abstraction is able to exploit the advantages of each type of code.

When stimuli are represented by populations of active neurons, each neuron may be involved in representing many stimuli and there may be considerable overlap in the populations of cells encoding different stimuli. Distributed codes thus suffer from interference between representations which can cause difficulties in distinguishing stimuli and in learning (Földiák and Young, 1995). In contrast, when a stimulus is represented by an individual neuron there is no interference between representations (Földiák and Young, 1995) and it is a trivial task to associate this neuron with any desired response. Hence, recoding information that is initially distributed into activations at more specialized individual neurons can make learning tractable by transforming complex, relational, tasks into simpler, statistical, problems over the recoded data (Clark and Thornton, 1997; Spratling, 1999).

Neurons which are selective for abstract features of the environment can act as 'virtual sensors' (Thornton, 1996a, 1997) or 'feature detectors' (Barlow, 1990, 1995) for specific stimuli which might otherwise be difficult to identify based on low-level sensory data. However, any information that is represented by an individual neuron at one level in the hierarchy must be simultaneously represented in a more distributed code at lower levels. These distributed representations do not simply provide the inputs to neurons tuned to more abstract stimuli but have a vital computational role. Such codes provide generalization and have much greater, long-term, storage capacity (Földiák and Young, 1995). Furthermore, many stimuli will not be represented by a specialized neuron at all, and will only be represented by a distributed code. It would be equally impractical to represent every possible stimulus

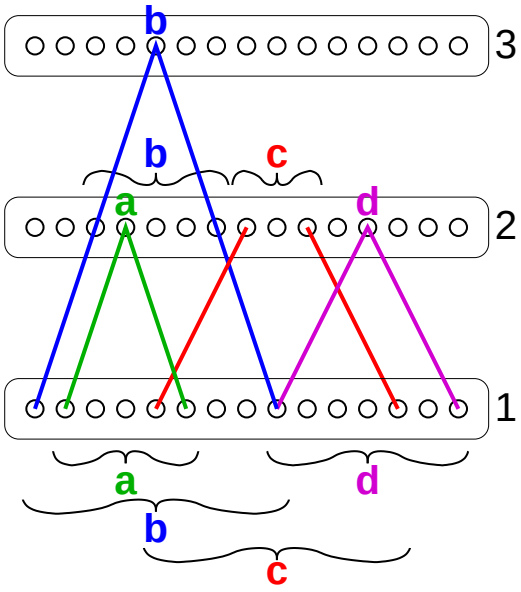

Figure 12: Coding information in a neural hierarchy. Neurons in three regions are illustrated representing four stimuli. Each stimulus is represented by a different population of neurons in the first region. In the second region two of the patterns ('a' and 'd') are represented by individual neurons which are selective to the corresponding patterns of activity in the previous region. The other stimuli ('b' and 'c') are represented in the second region by populations of neurons. Pattern 'b' is represented in the third region by an individual neuron which is selective to activity across the population of neurons representing 'b' in the second region.

using a dedicated cell as it would be to represent every possible stimulus using a distributed representation across cells tuned to features at the lowest-level (Feldman, 1990; Tsotsos, 1995; Wilson, 1991). The appropriate level of abstraction will vary between tasks but by recoding information the cortex can utilize the most appropriate representations for performing different tasks on the same data.

4.2 Learning Representations

Some stimuli will only ever be encoded using a distributed pattern of activity over a population of cells. Other information will not only be represented in this way but may also (and simultaneously) be represented by an individual neuron tuned specifically to that stimulus. Which stimuli do become represented by individual neurons, at some level in the processing hierarchy, will depend on experience and the significance of those stimuli to the animal and the tasks it needs to perform (Logothetis, 1998; Barlow, 1995; Page, 2000; Wallis and Bülthoff, 1999; Sigala and Logothetis, 2001). The selectivity of individual cells is subject to change through learning (Kobatake et al., 1998; Logothetis, 1998; Desimone, 1996). Extended experience leads to the response properties of certain cells becoming increasingly selective to the training stimulus (Rainer and Miller, 2000). In addition, there is a reduction in the size of the population of cells that become active in response to the familiar stimulus (Rainer and Miller, 2000) and improved discrimination performance is correlated with a smaller active population of neurons (Desimone, 1996). In addition, the cortical regions involved in task performance may change with expertise (Gauthier et al., 2000; Sigman and Gilbert, 2000; Walsh et al., 1998) such that there are fewer active regions once a task is automated than when it is novel (Petersen et al., 1998).

The representation of both abstract (Logothetis, 1998) and simple (Karni, 1996) stimuli can change with experience. Hence, at all levels in the perceptual information processing hierarchy the selectivity of neurons can be shaped by past experience (Mountcastle, 1998; Gilbert, 1996). Individual neurons learn to become selective for features of the input environment to which they are exposed during training. To do so the neurons need to adjust their weights to become selective to stimuli that recur within the training data. Such neurons become selective to features that are useful for representing stimuli. Furthermore, such features are likely to correspond to meaningful categories since the co-occurrence of patterns of inputs, which recur with greater probability than would be expected by chance, are likely to have some common underlying cause (Földiák, 1990; Barlow, 1990, 1994; Edelman and Duvdevani-Bar, 1995). The environment experienced by, and which influences learning at, each neuron is the pattern of pre-synaptic activity to which it is exposed. This 'environment' is thus different

for neurons in different cortical regions. Sensory stimuli generate patterns of activations in sensory nerve cells. These neural activations are themselves patterns of activity which can in turn be represented by further neural activations. Hence, the same learning mechanisms which develop useful codings for sensory data can develop representations of neural data, and so on hierarchically, such that at each stage neurons become tuned to ever more abstract features of the environment (Clark and Thornton, 1997; Thornton, 1997, 1996a,b). However, this does not imply that at the highest levels of the cortical hierarchy only local representations are used since at any level there will always be stimuli that are even more abstract than the stimulus selectivities of the individual neurons and which thus give rise to distributed representations.

The most accurate and complete representation of the environment will be provided by the raw sensory data. Information can only be lost in translating this data into new representational formats (deCharms and Zador, 2000). However, as discussed above, computational advantages can be gained from such recoding and learning enables such additional computational properties to be utilized for certain important, recurring, or behaviorally relevant stimuli. In addition, past experience (stored in the synaptic weights) can modify and augment the available data as it is transformed (via those weights). Knowledge is stored by means of synaptic modifications in the neural circuits which will use this information (McClelland et al., 1995; Karni, 1996) and neurons respond to stimuli based on the selectivity of their synaptic weights. Hence, representations are based on past experience and stimuli are interpreted in terms of the knowledge stored in the synaptic weights (Gosselin and Schyns, 2002; Mountcastle, 1998). Past experience thus plays a causal role in processing information.

Note that the response properties of cortical cells are much more complex than the caricature that has been presented here. The activities of cells are not purely driven by bottom-up, perceptual, processes but are strongly influenced by other factors such as task requirements, context, attention and expectation (Lamme et al., 1998). However, the response of a cell is still determined by the pattern of pre-synaptic activation it receives and hence the above arguments are still valid.

5 Conclusions

In communication theory, information is considered to be encoded as a message defined over a number of discrete symbols. Messages of varying lengths can result when different information is encoded over the same set of symbols. Messages of varying lengths can also occur when the same information is encoded using different sets of symbols. It makes little sense to stipulate that all messages must contain single symbols, or that they must contain multiple symbols, without reference to the information to be encoded and to the symbols available for the encoding. However, this is exactly what is happening in the debate about whether neural codes are local or distributed (or of any other predetermined density). Such arguments are especially misguided given that it has been established that neurons in different cortical regions are selective to different properties. In systems (like the ventral visual pathway) where networks of neurons are organized into a hierarchy of processing stages, information represented by the response of many neurons at one level may be represented by the response of individual neurons at a higher level. Hence, local and distributed codes coexist across different cortical regions. Furthermore, local and distributed codes coexist within each cortical region since for any population of neurons, with a diversity of tuning properties, there will be certain stimuli that activate multiple neurons, other stimuli that activate individual neurons, and still others that cause no response from any of the neurons.

Both local and distributed coding schemes are predicated on the same representational mechanism in which neurons denote information based on the selectivity of their synaptic weights. For such a representational mechanism the number of neurons which become active in response to the current stimulus should depend on the tuning properties of the neurons. Whatever the tuning properties of the neurons there will be certain stimuli that generate a local code and other stimuli that ought to generate a distributed code. Hence, any connectionist cognitive model should produce both local and distributed representations under different circumstances. Furthermore, a neural network algorithm should be capable of learning different encodings under different task demands. The mechanism of competition used between the nodes in a self-organising neural network determines if these requirements are met. A large and influential class of algorithms, based on one mechanism of competition, are capable of learning only one, pre-determined, coding scheme, irrespective of the task. Furthermore, several of these models are also restricted to generating representations encoded using one scheme, irrespective of the stimulus. We have therefore suggested that these algorithms are computationally deficient and that they provide incomplete models of the representational abilities of the cortex. We have demonstrated that an alternative mechanism of competition enables nodes to learn stimulus selectivities that depend on the training environment, and to respond to stimuli based on these tuning properties. Hence, this alternative neural network architecture learns to encode information in the most appropriate form for the task on which it is trained and can exploit the advantages of both local and distributed coding schemes. We therefore suggest that this algorithm provides a better model of the representational

capacities of cortical circuits.

Appendix: Implementation Details

A.1 Activation

Pre-integration lateral inhibition causes each node to inhibit other nodes from responding to the same inputs. Hence, if a node is active and it has a strong synaptic weight to a certain input then it should inhibit other nodes from responding to that input. A simple implementation of this idea, for a two-node network (such as that shown in figure 3(b)), would be:

$$y_1 = \sum_{i=1}^{m} \left(w_{i1}x_i - \alpha w_{i2}y_2\right)^+$$

$$y_2 = \sum_{i=1}^{m} \left(w_{i2}x_i - \alpha w_{i1}y_1\right)^+ .$$

Where y_j is the activation of node j, w_{ij} is the synaptic weight from input i to node j, x_i is the activation of input i, m is the total number of inputs, α is a scale factor controlling the strength of lateral inhibition, and $(v)^+$ is the positive half-rectified value of v. These simultaneous equations can be solved iteratively. To help ensure that a steady-state solution is reached it has been found useful to gradually increase the value of α at each iteration from an initial value of zero. Physiologically, this regime might correspond to a delay in the build up of lateral inhibition due to additional propagation delays via inhibitory interneurons. An equivalent implementation of post-integration lateral inhibition would be:

$$y_1 = \sum_{i=1}^{m} \left(w_{i1}x_i\right) - \alpha q_{21}y_2$$

$$y_2 = \sum_{i=1}^{m} \left(w_{i2}x_i\right) - \alpha q_{12}y_1.$$

Where q_{kj} is the lateral weight from node k to node j. Note that for pre-integration lateral inhibition rectification is applied to the inhibited inputs prior to summation. This is essential otherwise pre-integration lateral inhibition would be mathematically equivalent to post-integration lateral inhibition. For example, without rectification the equation given above for pre-integration lateral inhibition to node 1 would become $y_1 = \sum_{i=1}^{m} \left(w_{i1}x_i - \alpha w_{i2}y_2\right)$ which can be rewritten as $y_1 = \sum_{i=1}^{m} \left(w_{i1}x_i\right) - \alpha y_2 \sum_{i=1}^{m} w_{i2}$ which is equivalent to the equation for post-integration lateral inhibition when $\sum_{i=1}^{m} w_{i2} = q_{21}$. In section A.3 we suggest that pre-integration lateral inhibition is achieved in the cortex via inhibitory contacts on the dendrites of excitatory cells. Due to the nonlinear response properties of dendrites (Mel, 1993, 1994; Koch et al., 1983; Shepherd and Brayton, 1987) such dendritic inhibition will have effects confined to the local region of the dendritic tree and little or no impact on excitatory inputs to other branches of the dendrite. This nonlinearity is achieved in our model by using rectification.

While the equations presented above provide a simple illustration of pre-integration lateral inhibition a more complex formulation is required for application to neural networks containing arbitrary numbers of nodes (n). In a network containing more than two nodes each input could be inhibited by multiple lateral connections. What strength of inhibition should result from multiple nodes? One possibility would be to sum the effects from all the inhibitory connections. However, simply changing the number of nodes in such a network could have a significant effect on the total inhibition. To ensure that inhibition is not a function of n we inhibit each node by the maximum value of the inhibition generated by all the nodes. Hence, the activation of each node is calculated as:

$$y_j = \sum_{i=1}^{m} \left(w_{ij}x_i - \alpha \max_{\substack{k=1 \\ (k \neq j)}}^{n} \left\{ w_{ik}y_k \right\} \right)^+ .$$

Note that nodes do not inhibit their own inputs.

If two (or more) nodes have similar weights, and are strongly activated by the current stimulus, then those nodes should compete for the right to represent that stimulus. However, as nodes inhibit each other their activation values decrease which in turn reduces the strength of inhibition (y_k decreases in the above equation). Such a decrease in the strength of lateral inhibition is counter-productive and may prevent competition proceeding to a conclusion. Furthermore, early in training nodes tend to have small, undifferentiated, weights and hence small activation values. This results in weak lateral inhibition that permits all nodes to respond to each input pattern. To ensure that lateral inhibition remains effective when nodes inhibit each other and to ensure that lateral inhibition

is equally effective at the start of training, as at the end, the strength of inhibition is normalized as follows: the lateral weight is divided by the maximum lateral weight originating from the inhibiting node and the inhibiting node's activity is divided by the maximum activity of all the nodes in the network:

$$y_j = \sum_{i=1}^{m} \left(w_{ij} x_i - \alpha \max_{\substack{k=1 \\ (k \neq j)}}^{n} \left\{ \frac{w_{ik}}{\max_{l=1}^{m} \{w_{lk}\}} \frac{y_k}{\max_{l=1}^{n} \{y_l\}} \right\} \right)^+ .$$

Normalizing the strength of lateral inhibition ensures that there is strong competition resulting in a differentiation in activity values and more rapid activity-dependent learning from the start of training. Normalization also results in the strength of inhibition being constrained to be in the range zero to one. However, this form of inhibition would have a greater affect on a stimulus presented at low contrast (using smaller values of x_i) than on the same stimulus presented at high contrast. Tolerance to changes in stimulus contrast is provided by modifying the equation as follows:

$$y_j = \sum_{i=1}^{m} w_{ij} x_i \left(1 - \alpha \max_{\substack{k=1 \\ (k \neq j)}}^{n} \left\{ \frac{w_{ik}}{\max_{l=1}^{m} \{w_{lk}\}} \frac{y_k}{\max_{l=1}^{n} \{y_l\}} \right\} \right)^+ .$$

This equation can be re-written as:

$$y_j = \sum_{i=1}^{m} w_{ij} X_{ij} \tag{1}$$

Where X_{ij} is the input activation from source i to node j after inhibition:

$$X_{ij} = x_i \left(1 - \alpha \max_{\substack{k=1 \\ (k \neq j)}}^{n} \left\{ \frac{w_{ik}}{\max_{l=1}^{m} \{w_{lk}\}} \frac{y_k}{\max_{l=1}^{n} \{y_l\}} \right\} \right)^+ . \tag{2}$$

This formulation was used of produce all the results presented in this paper. The value of α was increased from zero to six in steps of 0.25. Activation values generally reached a steady-state at lower alpha, in which case competition was terminated prior to α reaching its maximum value. The change in the value of α between iterations was found to be immaterial to the final activation values provided it was less than 0.5.

A.2 Learning

Networks were trained using unsupervised learning. Weights were initially set all equal to $\frac{1}{m}$ (we describe such nodes as 'uncommitted'), although the algorithm works equally well with randomly initialized weights. In order to cause differentiation of the receptive fields, the activations at each iteration during the competition were modified by a noise term, such that $y_j \rightarrow y_j(1 + \rho)$. Noise values, ρ, were exponentially distributed positive real numbers with a mean of 0.001. Since the magnitude of the noise is small it has no effect on competition once nodes have begun to learn preferences for distinct input patterns.

Synaptic weights were modified at completion of the competition phase (*i.e.*, using the final values for y_j found after iteration of equation 1). The following learning rule was employed:

$$\Delta w_{ij} = \beta \ \frac{(x_i - \bar{x})}{\sum_{k=1}^{m} x_k} \ \frac{(y_j - \bar{y})^+}{\sum_{k=1}^{n} y_k}. \tag{3}$$

Where $\bar{x}$ is the mean of the input activations (*i.e.*, $\bar{x} = \frac{1}{m} \sum_{i=1}^{m} x_i$), and $\bar{y}$ is the mean of the output activations (*i.e.*, $\bar{y} = \frac{1}{n} \sum_{j=1}^{n} y_j$), and β is a parameter controlling the learning rate ($\beta = 1$ was used in all the simulations presented in this text). Following learning, synaptic weights were clipped at zero such that $w_{ij} = (w_{ij})^+$ and were normalized such that $\sum_{i=1}^{m} (w_{ij})^+ = 1$.

This learning rule encourages each node to learn weights selective for a set of coactive inputs. This is achieved since when a node is more active than average it increases its synaptic weights to active inputs and decreases its weights to inactive inputs. Hence, only sets of inputs which are consistently coactive will generate strong afferent weights. In addition, the learning rule is designed to ensure that nodes can represent stimuli which share input features in common (*i.e.*, to allow the network to represent overlapping patterns). This is achieved by rectifying the post-synaptic term of the rule so that no weight changes occur when the node is less active than average. If learning was not restricted in this way whenever a pattern was presented all nodes which represented patterns with overlapping features would reduce their weights to these features. Our learning rule is similar to the instar rule

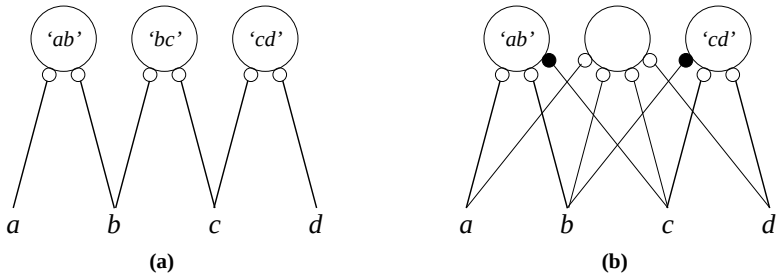

Figure 13: An illustration of a the role of negative weights. Nodes are shown as large circles, excitatory synapses as small open circles and inhibitory synapses as small filled circles. (a) Each node represents one on three input patterns 'ab', 'bc', and 'cd'. (b) Individual nodes are selective to patterns 'ab' and 'cd' and the network would represent pattern 'bc' by partially activating both of these nodes. If pattern 'bc' re-occurs sufficiently frequently negative afferent weights develop which enable the uncommitted node to compete to respond to this pattern.

(Grossberg, 1976, 1978), except that only nodes that are more active than average have their weights modified. Hence, post-synaptic activation exceeding a threshold is required before synaptic modification is enabled. The learning rule is also similar to the covariance rule (Sejnowski, 1977; Grzywacz and Burgi, 1998), except the post-synaptic term is only allowed to be greater than or equal to zero. Hence, sub-threshold post-synaptic activity does not result in any changes in synaptic strength. Normalization by the total of the input and output activities helps to ensure that each stimulus contributes equally to learning and that the total weight change (across all nodes) at each iteration is similar. Division by zero errors were avoided by learning only when the maximum input and output activities exceeded an arbitrary threshold (0.1).

The magnitudes of the lateral weights have identical values to the corresponding afferent weights (figure 3(c)). In a more biologically plausible version of the model separate lateral weights could be learnt independently. This would be possible since weights that have identical values also have identical (but reversed) pre- and post-synaptic activation values. For example, in figure 3(c) the afferent weight w_{12} connects input 1 to node 2 and a lateral weight of the same magnitude connects node 2 to input 1 (at node 1). Hence, by using similar activity-dependent learning rules, and identical initial values, appropriate lateral weights could be learnt.

Nodes learn to respond to specific features within the input space. In order to represent stimuli which contain multiple features the activation function allows multiple nodes to be active simultaneously. With overlapping patterns this could allow the same stimulus to be represented in multiple ways. For example, consider a simple network receiving input from four sources (labelled 'a', 'b', 'c' and 'd') which has the task of representing three patterns: 'ab', 'bc', and, 'cd'. Individual nodes in a network may learn to become selective to each of these patterns (figure 13(a)). Each individual pattern would thus be represented by the activity of a single node. However, it would also be possible for a network in which nodes had learnt patterns 'ab' and 'cd' to respond to pattern 'bc' by partially activating each of the nodes selective to 'ab' and 'cd'. Pattern 'bc' could be the result of the coactivation of partially occluded versions of patterns 'ab' and 'cd'. However, if 'bc' occurs with similar frequency to the other two patterns then it would seem more likely that 'bc' is an independent input pattern that should be represented in a similar way to the other patterns (*i.e.*, by the activation of a specific node tuned to that pattern). To ensure that in such situations the network learns all input patterns it is necessary to introduce a second learning rule:

$$\Delta w_{ij}^{-} = \beta^{-} \frac{(2X_{ij} - x_i)^{-}}{\sum_{k=1}^{m} x_k} \frac{(y_j - \bar{y})}{\sum_{k=1}^{n} y_k}. \tag{4}$$

Where X_{ij} is the input activation from source i to node j after inhibition (equation 2), β^{-} is a parameter controlling the learning rate ($\beta^{-} = 1$ was used in all the simulations presented in this text) and $(v)^{-}$ is the negative half-rectified value of v. Negative weights were constrained such that $0 \geq \sum_{i=1}^{m} (w_{ij})^{-} \geq -1$. This learning rule is only applied to synapses which have a weight of zero (or less than zero) caused by application of the learning rule given in equation 3 (or prior application of equation 4). The application of this second learning rule in the situation described above, where a network has learnt patterns 'ab' and 'cd' but not pattern 'bc', would result in negative weights being generated as shown in figure 13(b). Negative weights are generated when a node is active and inputs, which are not part of the nodes' preferred pattern, are inhibited. This can only occur when multiple nodes are coactive. If the pattern, to which this set of coactive nodes are responding, re-occurs then the negative weights will grow. When the negative weights are sufficiently large the response of these nodes to this particular pattern will be inhibited, enabling an uncommitted node to successfully compete to represent this pattern. On the

other hand, if the pattern, to which this set of coactive nodes are responding, is just due to the coactivation of independent input patterns then the weights will return towards zero on subsequent presentations of these patterns in isolation.

For programming convenience we allow a single synapse to take either excitatory or inhibitory weight values. In a more biologically plausible implementation two separate sets of afferent connections could be used: the excitatory ones being trained using equation 3 and the inhibitory ones being trained using a rule similar to equation 4. Note that the simplification we have used, that employs one set of weights to depict both excitatory and inhibitory synapses, does not result in any modification to the activation function (equation 1): inhibitory afferents can themselves be inhibited and the use of identical lateral weights is unlikely to cause lateral facilitation due to the $\max$ operation that is applied.

A.3 Plausibility

With pre-integration lateral inhibition learning is reliable and fast such that results for a standard benchmark problem compare very favorably with other unsupervised learning algorithms (Spratling and Johnson, 2002). Calculating node activations is tractable, requiring a polynomial-time algorithm, that is slightly slower than for post-integration lateral inhibition (computational complexity increases from $O(nm + n^2)$ for post-integration lateral inhibition to $O(mn^2)$ for pre-integration lateral inhibition).

The algorithm is also biologically plausible. Lateral inhibition between cortical cells is known to play an important role in determining the receptive field properties of those cells (Eysel et al., 1998; Jagadeesh, 2000). Our architecture requires inhibition specific to particular synaptic inputs. The vast majority (93%) of inhibitory synapses targeting neocortical pyramidal cells terminate on the dendrites (DeFelipe and Fariñas, 1992; Fiala and Harris, 1999). Such contacts have relatively little impact on excitatory inputs more proximal to the cell body or on the action of synapses on other branches of the dendritic tree but do have strong inhibitory effects on inputs within the same dendritic branch that are more distal to the site of inhibition (Spruston et al., 1999; Borg-Graham et al., 1998; Koch et al., 1983; Rall, 1964; Segev, 1995; Koch and Segev, 2000). Hence, they can potentially selectively inhibit specific synapses or groups of excitatory inputs. Related synapses cluster together within the dendritic tree so that local operations are performed by multiple, functionally distinct, dendritic subunits before integration at the soma (Koch and Segev, 2000; Segev and Rall, 1998; Segev, 1995; Häusser et al., 2000; Häusser, 2001; Mel, 1993, 1994). Dendritic inhibition could thus act to 'block' the output from individual functional compartments. If, for modeling convenience, all the synapses contributing to a dendritic compartment are grouped together as a single input then our algorithm can be viewed as a model of dendritic inhibition.

The idea embodied in our model is that pyramidal cells inhibit the activity of dendritic compartments in other pyramidal cells within the same cortical area. Our architecture thus also requires that dendritic inhibition be caused by other pyramidal cells. This claim is anatomically plausible since it has been shown that cortical pyramidal cells innervate inhibitory cell types which in turn form synapses on the dendrites of pyramidal cells (Buhl et al., 1997; Tamas et al., 1997). Further support for this model comes from physiological data which indicates that dendritic inhibition plays a role in determining the tuning properties of pyramidal cells. Wang et al. (2000) report that blockade of GABAergic (inhibitory) synapses alters the selectivity of cells in monkey inferotemporal cortex (area TE) in a way that suggests that inhibition is specific to particular excitatory inputs.

As with models of post-integration lateral inhibition we have simplified reality by assuming that the role of inhibitory cells can be approximated by direct inhibitory weights from excitatory cells. A more biologically accurate model might include a separate inhibitory cell population. One potential implementation would allocate one inhibitory neuron to each input or dendritic compartment (i). This inhibitory cell would receive weighted connections from each excitatory node. It would perform a $\max$ operation over these inputs and apply the resulting inhibition equally to input i of every excitatory node. Using this scheme each input would be inhibited by a single synapse. In sub-cortical structures, were axo-axonal, terminal-to-terminal, synapses are common (Brown, 1991; Kandel et al., 1995), our model might have an even more direct biological implementation since an excitatory neuron can directly inhibit the input to another neuron via an excitatory synapse on the axon terminal of the input ('pre-synaptic inhibition'; Bowsher, 1970; Shepherd, 1990; Kandel et al., 1995).

Acknowledgements

This work was funded by MRC Research Fellowship number G81/512.

References

Barlow, H. B. (1990). Conditions for versatile learning, Helmholtz's unconscious inference, and the task of perception. *Vision Research*, 30:1561–71.

Barlow, H. B. (1994). What is the computational goal of the neocortex? In Koch, C. and Davis, J. L., editors, *Large-Scale Neuronal Theories of the Brain*, chapter 1. MIT Press, Cambridge, MA.

Barlow, H. B. (1995). The neuron doctrine in perception. In Gazzaniga, M. S., editor, *The Cognitive Neurosciences*, chapter 26. MIT Press, Cambridge, MA.

Borg-Graham, L. T., Monier, C., and Fregnac, Y. (1998). Visual input evokes transient and strong shunting inhibition in visual cortical neurons. *Nature*, 393(6683):369–73.

Bowsher, D. (1970). *Introduction to the Anatomy and Physiology of the Nervous System*. Blackwell, Oxford, UK.

Brown, A. G. (1991). *Nerve Cells and Nervous Systems: An Introduction to Neuroscience*. Springer-Verlag, London, UK.

Buhl, E. H., Tamas, G., Szilagyi, T., Stricker, C., Paulsen, O., and Somogyi, P. (1997). Effect, number and location of synapses made by single pyramidal cells onto aspiny interneurones of cat visual cortex. *The Journal of Physiology,*, 500(3):689–713.

Clark, A. and Thornton, C. (1997). Trading spaces: computation, representation and the limits of uninformed learning. *Behavioural and Brain Sciences*, 20(1):57–66.

Crick, F. and Asanuma, C. (1986). Certain aspects of the anatomy and physiology of the cerebral cortex. In Rumelhart, D. E., McClelland, J. L., and The PDP Research Group, editors, *Parallel Distributed Processing: Explorations in the Microstructures of Cognition. Volume 2: Psychological and Biological Models*, pages 333–71. MIT Press, Cambridge, MA.

deCharms, R. C. and Zador, A. (2000). Neural representation and the cortical code. *Annual Review of Neuroscience*, 23:613–47.

DeFelipe, J. and Fariñas, I. (1992). The pyramidal neuron of the cerebral cortex: morphological and chemical characteristics of the synaptic inputs. *Progress in Neurobiology*, 39(6):563–607.

Desimone, R. (1996). Neural mechanisms for visual memory and their role in attention. *Proceedings of the National Academy of Science USA*, 93:13494–9.

Edelman, S. and Duvdevani-Bar, S. (1995). Similarity, connectionism, and the problem of representation in vision. *Neural Computation*, 9:701–20.

Eggermont, J. J. (1998). Is there a neural code? *Neuroscience and Biobehavioral Reviews*, 22(2):355–70.

Eysel, U. T., Shevelev, I. A., Lazareva, N. A., and Sharev, G. A. (1998). Orientation tuning and receptive field structure in cat striate neurons during local blockade of intrcortical inhibition. *Neuroscience*, 84(1):25–36.

Feldman, J. A. (1990). Computational constraints on higher neural representations. In Schwartz, E. L., editor, *Computational Neuroscience*. MIT Press, Cambridge, MA.

Fiala, J. and Harris, K. (1999). Dendritic structure and spines. In Stuart, G., Spruston, N., and Häusser, M., editors, *Dendrites*, chapter 1, pages 1–34. Oxford University Press, Oxford, UK.

Field, D. J. (1994). What is the goal of sensory coding? *Neural Computation*, 6(4):559–601.

Földiák, P. (1989). Adaptive network for optimal linear feature extraction. In *Proceedings of the IEEE/INNS International Joint Conference on Neural Networks*, volume 1, pages 401–5, New York, NY. IEEE Press.

Földiák, P. (1990). Forming sparse representations by local anti-Hebbian learning. *Biological Cybernetics*, 64:165–70.

Földiák, P. (1991). Learning invariance from transformation sequences. *Neural Computation*, 3:194–200.

Földiák, P. and Young, M. P. (1995). Sparse coding in the primate cortex. In Arbib, M. A., editor, *The Handbook of Brain Theory and Neural Networks*, pages 895–8. MIT Press, Cambridge, MA.

Fukushima, K. (1988). Neocognitron: a hierarchical neural network capable of visual pattern recognition. *Neural Networks*, 1(2):119–30.

Gauthier, I., Skudlarski, P., Gore, J. C., and Anderson, A. W. (2000). Expertise for cars and birds recruits brain areas involved in face recognition. *Nature Neuroscience*, 3(2):191–7.

Gilbert, C. D. (1996). Plasticity in visual perception and physiology. *Current Opinion in Neurobiology*, 6(2):269–74.

Goodale, M. A. and Milner, A. D. (1992). Separate visual pathways for perception and action. *Trends in Neurosciences*, 15:20–5.

Gosselin, F. and Schyns, P. G. (2002). RAP: a new framework for visual categorization. *Trends in Cognitive Sciences*, 6(2):70–7.

Grossberg, S. (1976). Adaptive pattern classification and universal recoding, I: parallel development and coding of neural feature detectors. *Biological Cybernetics*, 23:121–134.

Grossberg, S. (1978). A theory of human memory: self-organisation and performance of sensory-motor codes,

maps, and plans. In Rosen, R. and Snell, F., editors, *Progress in Theoretical Biology*, volume 5, pages 233–374, London, UK. Academic Press.

Grossberg, S. (1987). Competitive learning: from interactive activation to adaptive resonance. *Cognitive Science*, 11:23–63.

Grzywacz, N. M. and Burgi, P.-Y. (1998). Towards a biophysically plausible bidirectional Hebbian rule. *Neural Computation*, 10:499–520.

Häusser, M. (2001). Synaptic function: dendritic democracy. *Current Biology*, 11(1):R10–2.

Häusser, M., Spruston, N., and Stuart, G. J. (2000). Diversity and dynamics of dendritic signalling. *Science*, 290(5492):739–44.

Hubel, D. H. (1988). *Eye, Brain, and Vision.* Scientific American Library.

Hummel, J. E. (2000). Localism as a first step toward symbolic representation. *Behavioural and Brain Sciences*, 23(4):480–1.

Jagadeesh, B. (2000). Inhibition in inferotemporal cortex: generating selectivity for object features. *Nature Neuroscience*, 3(8):749–50.

Kandel, E. R., Schwartz, J. H., and Jessell, T. M., editors (1995). *Essentials of Neural Science and Behavior*. Appleton & Lange, London, UK.

Karmiloff-Smith, A. (1994). Precis of Beyond Modularity - a developmental perspective on cognitive science. *Behavioural and Brain Sciences*, 17(4):693–706.

Karni, A. (1996). The acquisition of perceptual and motor skills: a memory system in the adult human cortex. *Cognitive Brain Research*, 5:39–48.

Kobatake, E., Wang, G., and Tanaka, K. (1998). Effects of shape-discrimination training on the selectivity of inferotemporal cells in adult monkeys. *Journal of Neurophysiology*, 80(1):324–30.

Koch, C., Poggio, T., and Torre, V. (1983). Nonlinear interactions in a dendritic tree: localization, timing, and role in information processing. *Proceedings of the National Academy of Science USA*, 80(9):2799–802.

Koch, C. and Segev, I. (2000). The role of single neurons in information processing. *Nature Neuroscience*, 3(supplement):1171–7.

Kohonen, T. (1997). *Self-Organizing Maps*. Springer-Verlag, Berlin.

Lamme, V. A. F., Supèr, H., and Spekreijse, H. (1998). Feedforward, horizontal, and feedback processing in the visual cortex. *Current Opinion in Neurobiology*, 8(4):529–35.

Logothetis, N. (1998). Object vision and visual awareness. *Current Opinion in Neurobiology*, 8(4):536–44.

Logothetis, N. and Sheinberg, D. L. (1996). Visual object recognition. *Annual Review of Neuroscience*, 19:577–621.

Markman, A. B. and Dietrich, E. (2000). In defense of representation. *Cognitive Psychology*, 40:138–71.

Marshall, J. A. (1995). Adaptive perceptual pattern recognition by self-organizing neural networks: context, uncertainty, multiplicity, and scale. *Neural Networks*, 8(3):335–62.

McClelland, J. L., McNaughton, B. L., and O'Reilly, R. C. (1995). Why the are complementary learning systems in the hippocampus and neocortex: insights from the successes and failures of connectionist models of learning and memory. *Psychological Review*, 102:419–57.

McGarry, K., Wermter, S., and MacIntyre, J. (1999). Hybrid neural systems: From simple coupling to fully integrated neural networks. *Neural Computing Surveys*, 2:62–93.

Mel, B. W. (1993). Synaptic integration in an excitable dendritic tree. *Journal of Neurophysiology*, 70(3):1086–101.

Mel, B. W. (1994). Information processing in dendritic trees. *Neural Computation*, 6:1031–85.

Mountcastle, V. B. (1998). *Perceptual Neuroscience: The Cerebral Cortex*. Harvard University Press, Cambridge, MA.

Murre, J. M. J., Phaf, R. H., and Wolters, G. (1992). CALM: categorizing and learning module. *Neural Networks*, 5(1):55–82.

Oja, E. (1989). Neural networks, principle components, and subspaces. *International Journal of Neural Systems*, 1:61–8.

Olshausen, B. A. and Field, D. J. (1996a). Emergence of simple-cell receptive properties by learning sparse code for natural images. *Nature*, 381:607–9.

Olshausen, B. A. and Field, D. J. (1996b). Natural image statistics and efficient coding. *Network: Computation in Neural Systems*, 7(2):333–9.

O'Reilly, R. C. (1998). Six principles for biologically based computational models of cortical cognition. *Trends in Cognitive Sciences*, 2(11):455–62.

O'Reilly, R. C. and Munakata, Y. (2000). *Computational Explorations in Cognitive Neuroscience: Understanding the Mind by Simulating the Brain*. MIT Press, Cambridge, MA.

Page, M. (2000). Connectionist modelling in psychology: a localist manifesto. *Behavioural and Brain Sciences*,

23(4):443–67.
Perrett, D. I. (1996). View-dependent coding in the ventral stream and its consequences for recognition. In Caminiti, R., Hoffmann, K.-P., Lacquaniti, F., and Altman, J., editors, *Vision and Movement Mechanisms in the Cerebral Cortex*, pages 142–51. HFSP, Strasbourg.
Perrett, D. I., Hietanen, J. K., Oram, M. W., and Benson, P. J. (1992). Organisation and functions of cells responsive to faces in the temporal cortex. *Philosophical Transactions of the Royal Society of London*, 335:23–30.
Petersen, S. E., Van Mier, H., Fiez, J. A., and Raichle, M. E. (1998). The effects of practice on the functional anatomy of task performance. *Proceedings of the National Academy of Science USA*, 95:853–60.
Rainer, G. and Miller, E. K. (2000). Effects of visual experience on the representation of objects in the prefrontal cortex. *Neuron*, 27:179–89.
Rall, W. (1964). Theoretical significance of dendritic trees for neuronal input-output relations. In Reiss, R. F., editor, *Neural Theory and Modeling*, pages 73–97. Stanford University Press, Stanford, CA.
Riesenhuber, M. and Poggio, T. (1999). Hierarchical models of object recognition in cortex. *Nature Neuroscience*, 2(11):1019–25.
Ritter, H., Martinetz, T., and Schulten, K. (1992). *Neural Computation and Self-Organizing Maps. An Introduction*. Addison-Wesley, Reading, MA.
Rumelhart, D. E. and Zipser, D. (1985). Feature discovery by competitive learning. *Cognitive Science*, 9:75–112.
Sanger, T. D. (1989). Optimal unsupervised learning in a single-layer linear feedforward neural network. *Neural Networks*, 2:459–73.
Segev, I. (1995). Dendritic processing. In Arbib, M. A., editor, *The Handbook of Brain Theory and Neural Networks*, pages 282–9. MIT Press, Cambridge, MA.
Segev, I. and Rall, W. (1998). Excitable dendrites and spines: earlier theoretical insights elucidate recent direct observations. *Trends in Neurosciences*, 21(11):453–60.
Sejnowski, T. J. (1977). Storing covariance with nonlinearly interacting neurons. *Journal of Mathematical Biology*, 4:303–21.
Shepherd, G. M., editor (1990). *The Synaptic Organization of the Brain*. Oxford University Press, Oxford, UK.
Shepherd, G. M. and Brayton, R. K. (1987). Logic operations are properties of computer-simulated interactions between excitable dendritic spines. *Neuroscience*, 21:151–66.
Sigala, N. and Logothetis, N. K. (2001). Visual categorization shapes feature selectivity in the primate temporal cortex. *Nature*, 415:318–20.
Sigman, M. and Gilbert, C. D. (2000). Learning to find a shape. *Nature Neuroscience*, 3(3):264–9.
Sirosh, J. and Miikkulainen, R. (1994). Cooperative self-organization of afferent and lateral connections in cortical maps. *Biological Cybernetics*, 71:66–78.
Spratling, M. W. (1999). *Artificial Ontogenesis: A Connectionist Model of Development*. PhD thesis, University of Edinburgh.
Spratling, M. W. and Johnson, M. H. (2002). Pre-integration lateral inhibition enhances unsupervised learning. *Neural Computation*, 14(9):2157–79.
Spruston, N., Stuart, G., and Häusser, M. (1999). Dendritic integration. In Stuart, G., Spruston, N., and Häusser, M., editors, *Dendrites*, chapter 10, pages 231–271. Oxford University Press, Oxford, UK.
Sun, R. (1995). Robust reasoning: integrating rule-based and similarity-based reasoning. *Artificial Intelligence*, 75(2):241–95.
Sun, R. (2001). Computation, reduction, and teleology of consciousness. *Cognitive Systems Research*, 1(4):241–9.
Tamas, G., Buhl, E. H., and Somogyi, P. (1997). Fast IPSPs elicited via multiple synaptic release sites by different types of GABAergic neurone in the cat visual cortex. *The Journal of Physiology,*, 500(3):715–38.
Tanaka, K. (1996). Representation of visual feature objects in the inferotemporal cortex. *Neural Networks*, 9(8):1459–75.
Thornton, C. (1996a). Re-presenting representation. In Peterson, D. M., editor, *Forms of Representation: An Interdisciplinary Theme for Cognitive Science*, pages 152–62. Intellect Books, Exeter, UK.
Thornton, C. (1996b). Unsupervised constructive learning. Technical Report CSRP 449, School of Cognitive and Computing Sciences, University of Sussex.
Thornton, C. (1997). Truth-from-trash learning and the mobot footballer. Technical Report CSRP 504, School of Cognitive and Computing Sciences, University of Sussex.
Thorpe, S. J. (1995). Localized versus distributed representations. In Arbib, M. A., editor, *The Handbook of Brain Theory and Neural Networks*, pages 549–52. MIT Press, Cambridge, MA.
Tsotsos, J. K. (1995). Behaviourist intelligence and the scaling problem. *Artificial Intelligence*, 75:135–60.
Ungerleider, L. G. (1996). What and where in the human brain? evidence from functional brain imaging studies. In Caminiti, R., Hoffmann, K.-P., Lacquaniti, F., and Altman, J., editors, *Vision and movement mechanisms in the cerebral cortex*, pages 23–30. HFSP, Strasbourg.

Ungerleider, L. G. and Mishkin, M. (1982). Two cortical visual systems. In Ingle, D. J., Goodale, M. A., and Mansfield, R. J. W., editors, *Analysis of Visual Behavior*, pages 549–86. MIT Press, Cambridge, MA.

van Gelder, T. (1999). Distributed vs. local representation. In Wilson, R. A. and Keil, F., editors, *The MIT Encyclopedia of the Cognitive Sciences*, pages 236–8. MIT Press, Cambridge, MA.

von der Malsburg, C. (1973). Self-organisation of orientation sensitive cells in the striate cortex. *Kybernetik*, 14:85–100.

Wallis, G. (1996). Using spatio-temporal correlations to learn invariant object recognition. *Neural Networks*, 9(9):1513–9.

Wallis, G. and Bülthoff, H. (1999). Learning to recognize objects. *Trends in Cognitive Sciences*, 3(1):22–31.

Walsh, V., Ashbridge, E., and Cowey, A. (1998). Cortical plasticity in perceptual learning demonstrated by transcranial magnetic stimulation. *Neuropsychologia*, 36(4):363–7.

Wang, Y., Fujita, I., and Murayama, Y. (2000). Neuronal mechanisms of selectivity for object features revealed by blocking inhibition in inferotemporal cortex. *Nature Neuroscience*, 3(8):807–13.

Wermter, S. and Sun, R. (2000). An overview of hybrid neural systems. In Wermter, S. and Sun, R., editors, *Hybrid Neural Systems*. Springer-Verlag, Heidelberg.

Wilson, S. W. (1991). The animat path to AI. In Meyer, J.-A. and Wilson, S. W., editors, *From Animals to Animats: Proceedings of the First International Conference on the Simulation of Adaptive Behaviour (SAB91)*, pages 15–21, Cambridge, MA. MIT Press.